Herbar Digital

Fachhochschule Hannover
University of Applied Sciences and Arts

Untersuchung von kommerzieller Software für den Einsatz im Projekt Herbar Digital

Prof. Dr. Karl-Heinz Steinke

Wintersemester 2007/08, Sommersemester 2008:

Projektmitarbeiter: Robert Dzido, Martin Gehrke,
Klaus Prätel

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Diagrammverzeichnis

Abkürzungsverzeichnis

Abb.	Abbildung
BGBM	Botanischer Garten/ Botanisches Museum
OCR	Optical Character Recognition
SDK	Software Development Kit

Kurzfassung

Die vorliegende Arbeit untersucht den möglichen Einsatz kommerzieller Software im Projekt Herbar Digital. Dabei werden zwei Kategorien unterschieden: OCR-Software und Barcodesoftware.

Von der ersten Kategorie gibt es eine Vielzahl käuflicher Programme auf dem Markt sowie auch einige kostenlose Freewareprogramme. Die Qualität ist jedoch sehr unterschiedlich, insbesondere fallen die Freewareprogramme stark ab. Es kristallisieren sich vier hochqualitative Programme heraus, die genau untersucht werden. Von diesen eignen sich zwei für das Projekt, wobei Omnipage 16 der Vorzug gegeben wird.

In der Kategorie der Barcodesoftware fiel die Wahl auf QS-Barcode 4.0, da sich OCR-Programme für das Lesen von Barcodes als ungeeignet erwiesen. Die anfängliche Erkennungsrate von 90% konnte durch eigene Verfahren zur Bildvorverarbeitung auf 100% gesteigert werden.

Abstract

The available work examines the possible use of commercial software in the project Herbar Digital. Two categories are differentiated: OCR software and bar code software.

From the first category there is a multiplicity of available programs on the market as well as some free programs. The quality between them is very differing; particularly the freeware programs are falling behind. Four high-quality programs emerge which are examined exactly. Two of these are suitable for the project, whereby Omnipage 16 the preference is given.

In the category of the bar code software the choice fell on QS-Barcode 4.0, since OCR programs for reading bar codes proved as unsuitable. The initial recognition rate of 90% could be increased by own procedures of picture preprocessing on 100%.

1 Untersuchung von kommerzieller Software für den Einsatz im Projekt Herbar Digital

1.1 Untersuchung von kommerzieller OCR-Software

Im Rahmen des Forschungsvorhabens „Herbar Digital“ soll OCR-Software bezüglich einer Einbindung in den Erkennungsvorgang untersucht werden.

Optische Schriftzeichenerkennung (Optical Character Recognition) oder verallgemeinert auch die Erkennung von Symbolen und Objekten ist einer der wesentlichsten Anwendungsbereiche der Mustererkennung.
Die OCR hat in den letzten Jahren erhebliche Fortschritte gemacht, ist aber noch immer Gegenstand intensiver Forschung.
Die heutigen, für normale Nutzer verfügbaren Programme sind in der Lage, hochqualitative Texte weitestgehend korrekt zu erkennen. Allerdings gibt es noch erhebliche Defizite, komplexe Dokumente mit aufwändigen Strukturierungen und Formatierungen, wie z.B. eingebetteten Bilddaten oder mit Tabellen, korrekt zu verarbeiten. Die vorliegenden Herbarien stellen eine komplexe Umgebung für die verschiedenen Texte (Druck- und Handschriften) dar, außerdem sind zusätzliche Objekte vorhanden wie Stempel, Barcode, Maßstäbe, Farbtafeln, Tüten usw.
Ziel dieser Untersuchung ist es, die aktuelle Software im Bereich der automatischen Erkennung von Text zu sichten und auf Tauglichkeit zu testen.
Es werden zunächst vier OCR-Programme hinsichtlich Erkennungsergebnis, Stapelverarbeitung, Dateiformat und Bedienoberfläche untersucht.

- Nuance OmniPage Professional 16
- ABBYY FineReader 9.0 Professional Edition
- I.R.I.S. ReadIris Pro 11 Corporate Edition
- Nuance TextBridge Pro 11

Abb. 1: OCR-Programme, die genau untersucht wurden

Die Freewareprogramme SimpleOCR, deren Texterkennung zurzeit nur Englisch und Französisch unterstützt, sowie Scan2PDF sind vom Funktionsumfang zu sehr eingeschränkt und werden nicht weiter betrachtet.

Im weiteren Verlauf des Berichtes werden die OCR-Programme wie folgt bezeichnet:

- ABBYY FineReader 8.0 Professional Edition: FineReader
- Nuance OmniPage Professional 16: OmniPage
- I.R.I.S. ReadIris Pro 11 Corporate Edition: ReadIris
- Nuance TextBridge Pro 11: TextBridge

1.1.1 Einleseformat

FineReader kann die vorliegenden Original-Bilder im TIF-Dateiformat problemlos einlesen. OmniPage, ReadIris und TextBridge sind nicht in der Lage, Bildgrößen von ca. 10.300 x 6.400 Pixel (Dateigröße ca. 200 Mbyte) zu verarbeiten. Deswegen werden diese Dateien auf ein Viertel der Bildgröße ins BMP-Format umgerechnet. Die Bildbreite und Bildhöhe werden halbiert, sodass sich die Dateigröße auf ungefähr 50 MByte reduziert. Bei dieser Dateikonvertierung gehen natürlich Bildinformationen verloren. Mit diesen reduzierten Bilddateien werden OmniPage, ReadIris und TextBridge für den Ergebnisvergleich konfrontiert.

1.1.2 Stapelverarbeitung, Batchbetrieb und SDK

Bei der Stapelverarbeitung werden mehrere Bilder dateiweise eingelesen, erkannt und mit Bezug auf den ursprünglichen Dateinamen gespeichert.
FineReader bietet ebenso wie OmniPage die Möglichkeit zur Stapelverarbeitung. ReadIris bietet auch die Möglichkeit der Stapelbearbeitung, allerdings werden die Erkennungsergebnisse der eingelesenen Dateien in einer Datei zusammengefasst. Mit TextBridge ist eine Stapelverarbeitung nicht möglich.
OmniPage bietet ebenfalls die Möglichkeit des Batchbetriebs. Hierbei können Arbeitsaufträge an das Programm erteilt werden, welche dann automatisch abgearbeitet werden.
Entwicklungsumgebungen (**S**oftware **D**evelopment **K**its) sind sowohl bei OmniPage als auch bei FineReader erhältlich. Somit lässt sich die Erkennungsmethode dieser Programme in selbst entwickelte Software einbinden.

1.1.3 Bedienoberfläche

Die Abbildungen (Abb.) 2 bis 5 zeigen die Bedienoberflächen von FineReader, OmniPage, ReadIris und TextBridge am Beispiel der Bilddatei „B_-W_19813 -01 0.tif“.
Alle Programme haben eine übersichtliche Bedienoberfläche. In Abb. 2 ist zu erkennen, dass FineReader eine spezielle Erkennung für Strichcodes aufweist (hellgrüne Markierung). FineReader, Omni-Page und TextBridge bieten die Möglichkeit der Nachbearbeitung der Erkennung in einem Texteditor. ReadIris bietet diese Möglichkeit nicht.

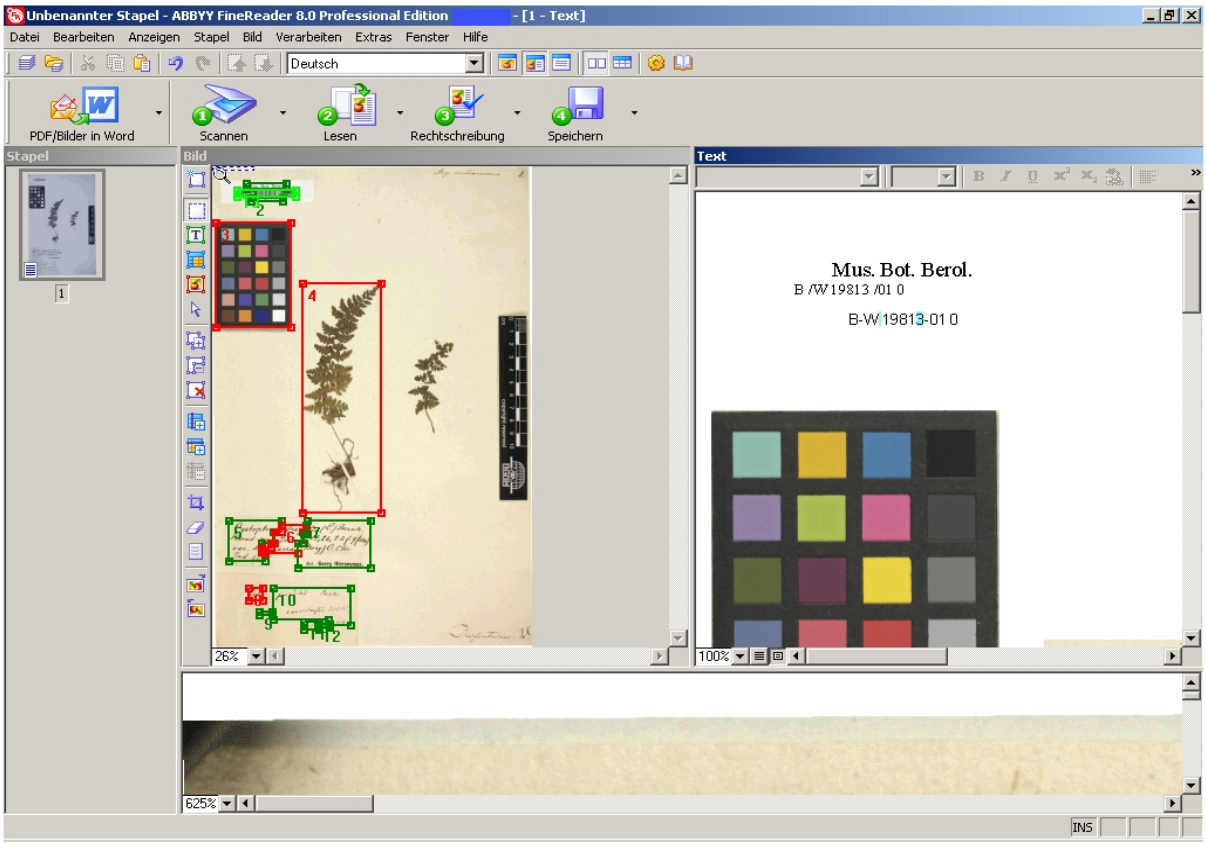

Abb. 2: **Bedienoberfläche von ABBYY FineReader 8.0 Professional**

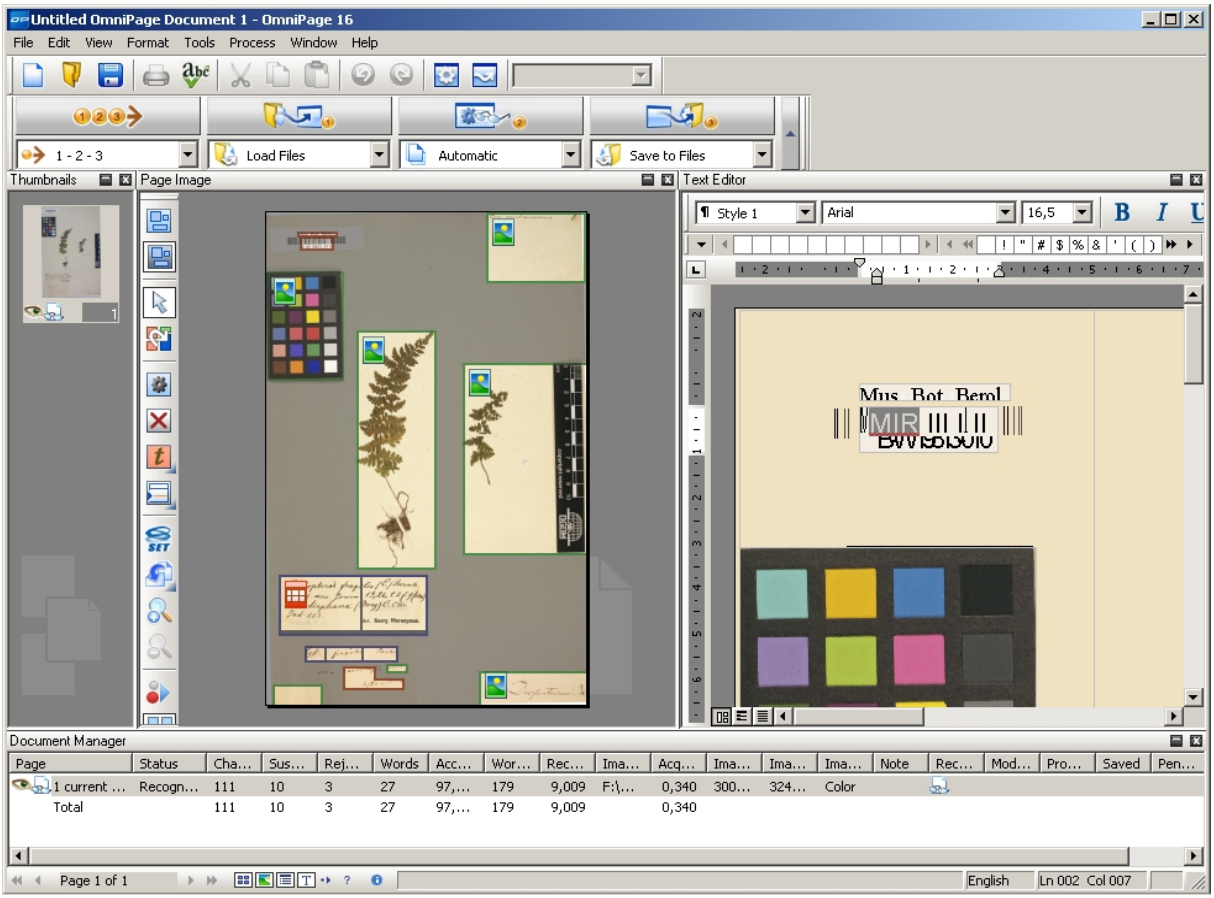

Abb. 3: Bedienoberfläche von Nuance OmniPage Professional 16

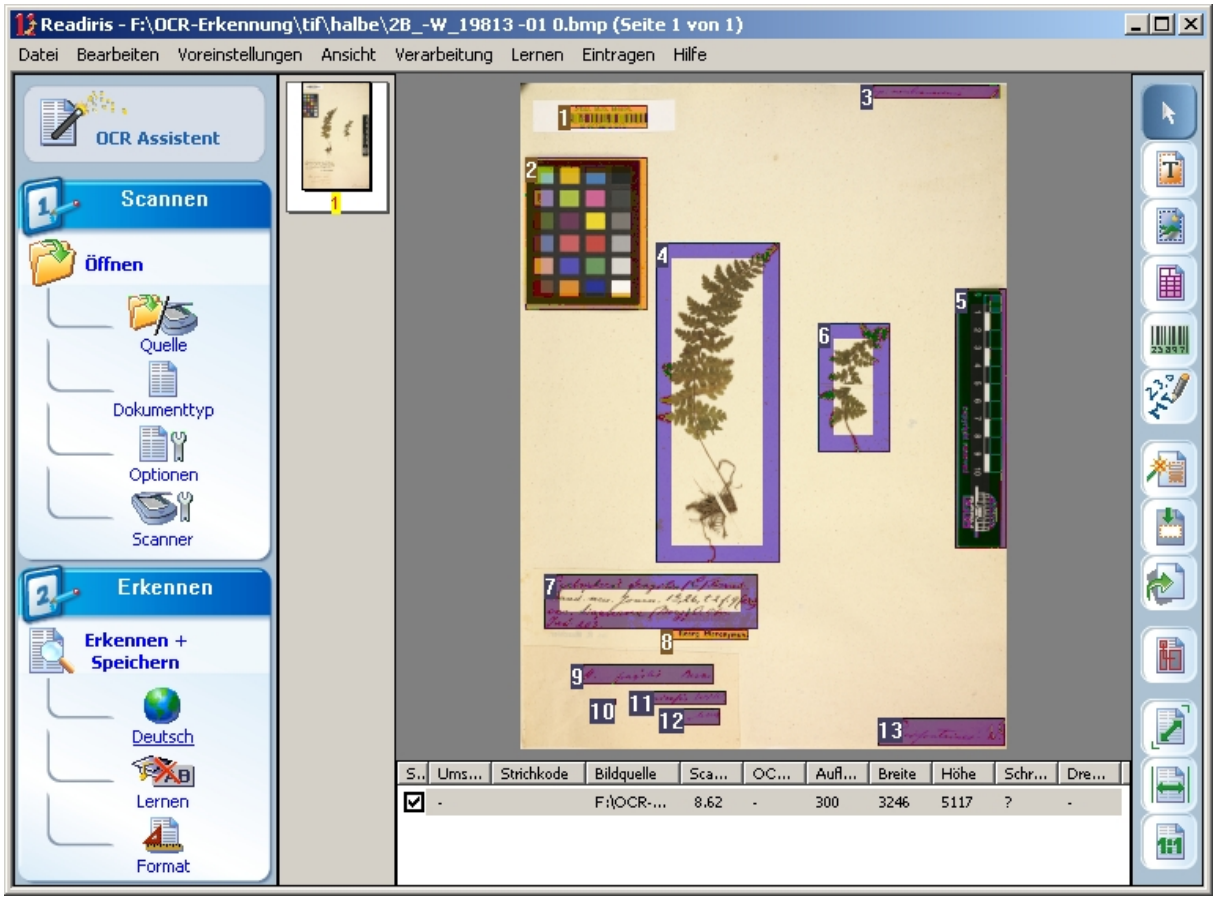

Abb. 4: Bedienoberfläche von I.R.I.S. Readlris Pro 11 Corporate Edition

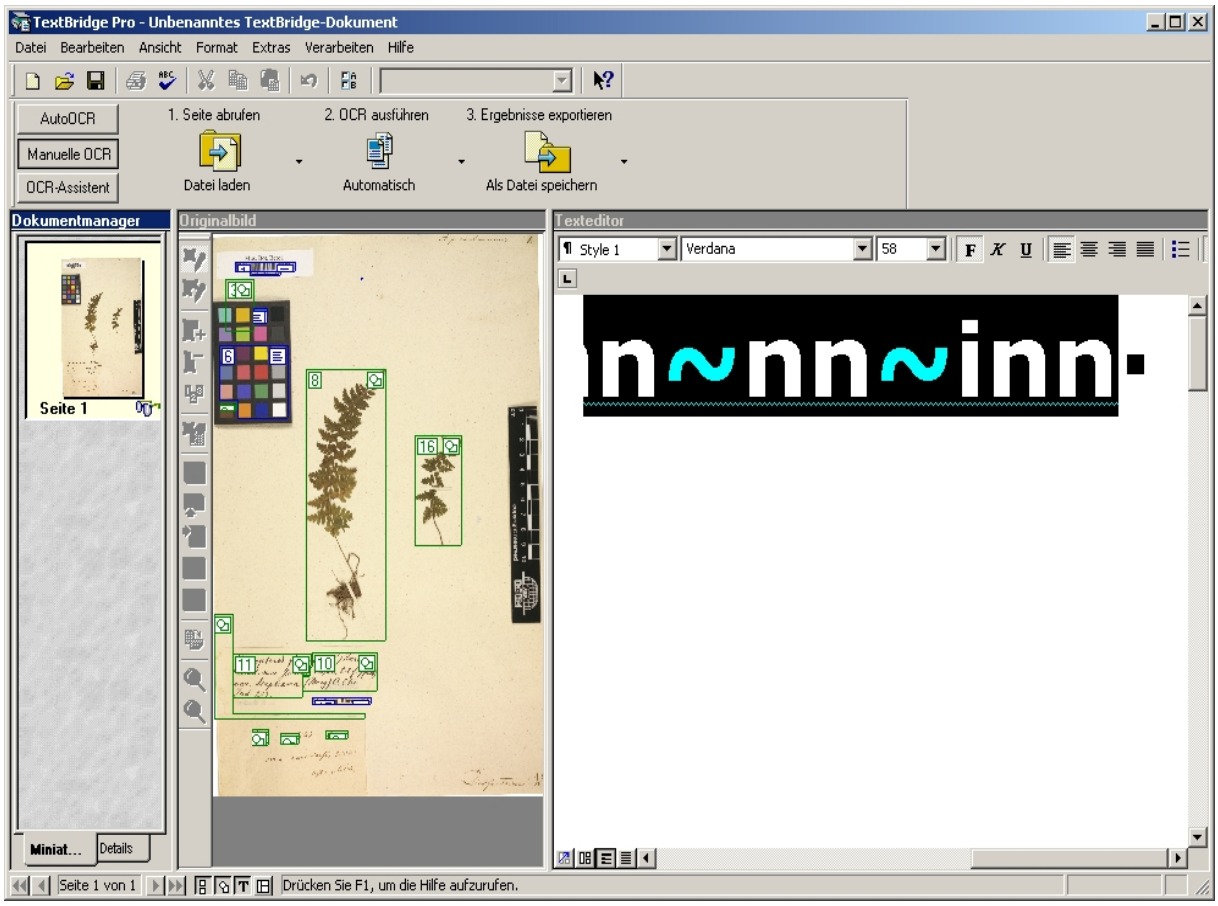

Abb. 5: Bedienoberfläche von Nuance TextBridge Pro 11

1.1.4 Vergleich der Programme

Es wurden die oben angegebenen OCR-Programme anhand einer Auswahl von Herbar-Bildern getestet und miteinander verglichen. Das Ergebnis der Untersuchung soll zeigen, welche Software für das Projekt am besten geeignet ist.
Da alle Hersteller eine Auflösung der Bilder von mindestens 300 dpi empfehlen, wurden solche Bilder aus den Originalvorlagen erstellt. Dazu wurden die Originalbilder (600 dpi) in jeder Richtung auf die Hälfte verkleinert. Das wurde auch deshalb notwendig, weil OmniPage nur Bilder mit maximal 8400 Zeilen verarbeiten kann.
Als Testobjekte dienten 15 Vorlagen mit einer Auflösung von 300 dpi mit insgesamt 4130 Zeichen und 757 Wörtern. Die Zeichen werden in verschiedene Kategorien eingeteilt wie Schreibschrift, Druckschrift, Sonderzeichen usw.
Nach den ersten Tests kristallisierten sich zwei Favoriten heraus: OmniPage und FineReader, die derzeit deutlich funktionsfähiger als die Anderen sind. Da sich bis auf diese zwei Programme alle als ungeeignet erwiesen, wurden die weiteren Tests nur mit OmniPage und FineReader durchgeführt.

Produkt	**OmniPage 16**	**FineReader 8.0**
Erfassungszeit/s	20,4	11,8
Erkennungszeit/s	28,8	34,2
Speicherzeit/s	14,2	20,2
Summe	63,4	66,2

Tabelle 1: Ausführungszeiten

Die Zeitangaben in der Tabelle 1 nennen die Dauer von Einlese- plus Erkennungsvorgang pro Dokument auf einem Rechner mit Pentiumprozessor, 1,8 GHz Takt und 1024 MByte Arbeitsspeicher.

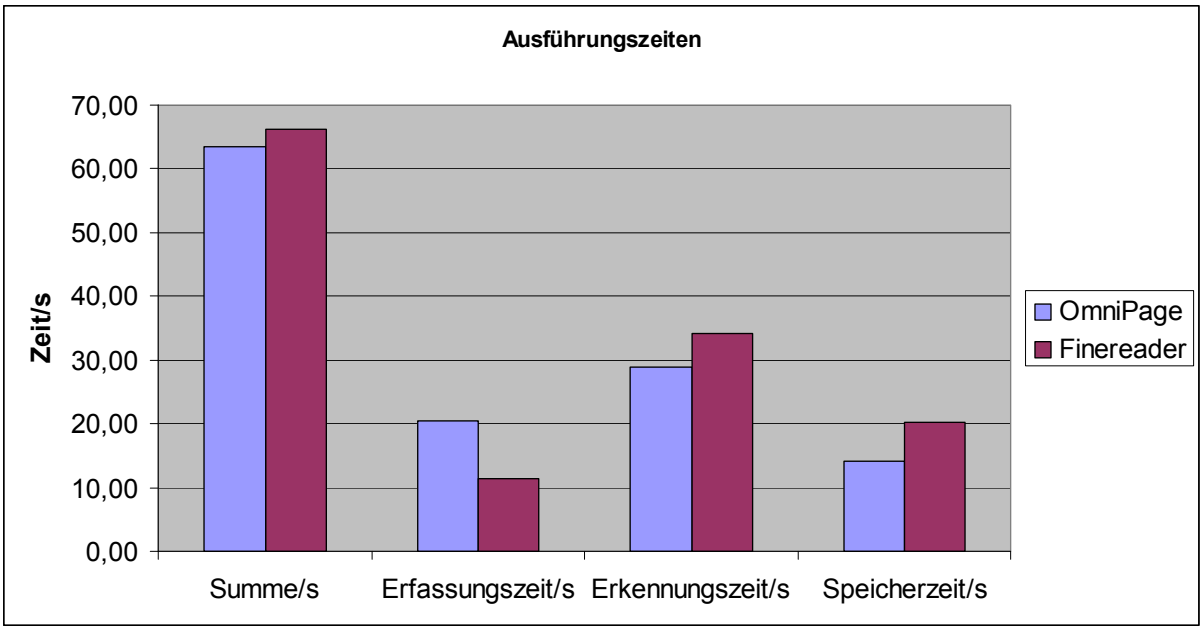

Diagramm 1: Ausführungszeiten

Neben dem Zeitfaktor spielen die Lesegenauigkeit und die Lokalisierung von Text eine noch wichtigere Rolle. Bei der Prüfung wurde unterschieden in

- Richtige Erkennung (Text wurde lokalisiert und Zeichen richtig gelesen)
- Falsche Erkennung (Text wurde lokalisiert und Zeichen falsch gelesen)
- Keine Erkennung (Text wurde nicht lokalisiert)

Bildernummer	Ges. Zeichen	Druckschrift	Handschrift	Sonderzeichen
		Ges. Zeichen	Ges. Zeichen	
2b_2011	229	139	67	23
2b_2012	160	97	45	18
2b_2106_Quer	205	121	65	19
2b_2698	271	190	63	26
2b_2699_Quer	244	151	69	24
10_2700_1	303	188	94	21
10_2710_1	266	181	57	28
10_2799_1	236	85	129	22
10_2800_1	158	91	61	13
10_89164_1	356	120	204	33
10_93680_1	384	335	49	38

10_93684_1	226	122	91	13
10_93692_1	374	142	193	38
10_93701_1	347	145	166	36
10_93703_1	371	142	198	31
Gesamt	4130	2249	1551	383
Prozent	100%	54%	37%	9%

Tabelle 2: Anzahl der drei verschiedenen Zeichenarten in 15 Testbildern

1.1.5 Erkennungsergebnisse OmniPage 16

Bildernummer	Ges. Wörter	Ges. Zeichen	Ges. Richtige Zeichen
2b_2011	40	229	63
2b_2012	29	160	29
2b_2106_Quer	38	205	56
2b_2698	36	271	173
2b_2699_Quer	46	244	148
10_2700_1	44	303	198
10_2710_1	43	266	144
10_2799_1	48	236	65
10_2800_1	38	158	98
10_89164_1	78	356	86
10_93680_1	46	384	329
10_93684_1	40	226	111
10_93692_1	85	374	95
10_93701_1	72	347	95
10_93703_1	74	371	84
Gesamt	757	4130	1774
Prozent	--------	100%	43%

Tabelle 3: Richtige Erkennung Gesamt OmniPage

Bildernummer	Druckschrift			
	Ges. Zeichen	Richtig	Falsch	Keine Erkennung
2b_2011	139	54	1	84
2b_2012	97	23	14	60
2b_2106_Quer	121	49	28	44
2b_2698	190	166	12	12
2b_2699_Quer	151	129	2	20
10_2700_1	188	175	13	0
10_2710_1	181	130	13	38
10_2799_1	85	48	21	16
10_2800_1	91	76	6	9
10_89164_1	120	74	9	46
10_93680_1	335	295	7	33
10_93684_1	122	102	11	9
10_93692_1	142	84	22	36
10_93701_1	145	79	26	40
10_93703_1	142	74	20	48
Gesamt	2249	1558	205	495
Prozent	100%	69%	9%	22%

Tabelle 4: Druckschrift-Erkennung OmniPage

Bildernummer	Handschrift			
	Ges. Zeichen	Richtig	Falsch	Keine Erkennung
2b_2011	67	1	2	64
2b_2012	45	2	1	42
2b_2106_Quer	65	0	2	63
2b_2698	63	0	0	63
2b_2699_Quer	69	5	28	36
10_2700_1	94	6	13	75
10_2710_1	57	5	38	14
10_2799_1	129	5	45	79

10_2800_1	61	12	15	34
10_89164_1	204	3	15	186
10_93680_1	49	0	11	38
10_93684_1	91	1	8	82
10_93692_1	193	3	38	152
10_93701_1	166	5	18	143
10_93703_1	198	3	32	163
Gesamt	1551	51	266	1234
Prozent	100%	3%	17%	80%

Tabelle 5:Handschrift-Erkennung OmniPage

Bildernummer	Sonderzeichen			
	Zeichen	Richtig	Falsch	Keine Erkennung
2b_2011	23	8	0	15
2b_2012	18	4	1	13
2b_2106_Quer	19	7	12	0
2b_2698	26	7	2	17
2b_2699_Quer	24	14	3	7
10_2700_1	21	17	3	1
10_2710_1	28	9	11	8
10_2799_1	22	12	4	2
10_2800_1	13	10	2	1
10_89164_1	33	9	5	19
10_93680_1	38	34	2	2
10_93684_1	13	8	2	3
10_93692_1	38	8	2	28
10_93701_1	36	11	2	23
10_93703_1	31	7	1	23
Gesamt	383	165	52	162
Prozent	100%	43%	14%	43%

Tabelle 6:Sonderzeichen-Erkennung OmniPage

1.1.6 Erkennungsergebnisse FineReader 8.0

Bildernummer	Ges. Wörter	Ges. Zeichen	Ges. Richtige Zeichen
2b_2011	40	229	145
2b_2012	29	160	115
2b_2106_Quer	38	205	16
2b_2698	36	271	215
2b_2699_Quer	46	244	13
10_2700_1	51	224	127
10_2710_1	35	264	2
10_2799_1	54	247	102
10_2800_1	38	158	103
10_89164_1	78	356	107
10_93680_1	46	384	364
10_93684_1	40	226	0
10_93692_1	85	374	117
10_93701_1	72	347	96
10_93703_1	74	371	4
Gesamt	762	4060	1526
Prozent	----------	100%	38%

Tabelle 7: Richtige Erkennung Gesamt FineReader

Bildernummer	Druckschrift			
	Ges. Zeichen	Richtig	Falsch	Keine Erkennung
2b_2011	139	123	5	11
2b_2012	97	54	14	29
2b_2106_Quer	121	11	73	37
2b_2698	190	187	3	0
2b_2699_Quer	151	0	41	110
10_2700_1	133	102	2	29
10_2710_1	186	0	0	186

10_2799_1	93	72	21	0
10_2800_1	91	88	2	1
10_89164_1	127	95	16	16
10_93680_1	349	337	12	0
10_93684_1	126	0	0	126
10_93692_1	145	113	32	0
10_93701_1	152	84	26	42
10_93703_1	149	3	1	145
Gesamt	2249	1269	248	732
Prozent	100%	56%	11%	33%

Tabelle 8: Druckschrift-Erkennung FineReader

Bildernummer	Handschrift			
	Ges. Zeichen	Richtig	Falsch	Keine Erkennung
2b_2011	67	5	25	89
2b_2012	45	46	4	21
2b_2106_Quer	65	0	0	75
2b_2698	63	5	41	23
2b_2699_Quer	69	1	15	59
10_2700_1	90	4	13	81
10_2710_1	58	2	6	50
10_2799_1	130	11	57	72
10_2800_1	61	15	14	32
10_89164_1	206	7	77	132
10_93680_1	49	0	0	49
10_93684_1	91	0	0	91
10_93692_1	193	7	0	44
10_93701_1	166	3	95	78
10_93703_1	198	0	11	197
Gesamt	1551	106	352	1093
Prozent	100%	7%	23%	70%

Tabelle 9: Handschrift-Erkennung FineReader

Bildernummer	Sonderzeichen			
	Zeichen	Richtig	Falsch	Keine Erkennung
2b_2011	23	17	0	5
2b_2012	18	15	1	2
2b_2106_Quer	19	5	6	8
2b_2698	26	23	3	0
2b_2699_Quer	24	13	6	5
10_2700_1	21	21	0	0
10_2710_1	28	0	1	24
10_2799_1	22	19	0	5
10_2800_1	13	9	0	4
10_89164_1	33	10	8	15
10_93680_1	38	37	1	0
10_93684_1	13	0	0	13
10_93692_1	38	35	5	0
10_93701_1	36	11	19	6
10_93703_1	31	1	0	30
Gesamt	383	216	50	117
Prozent	100%	56%	13%	31%

Tabelle 10: Sonderzeichen-Erkennung FineReader

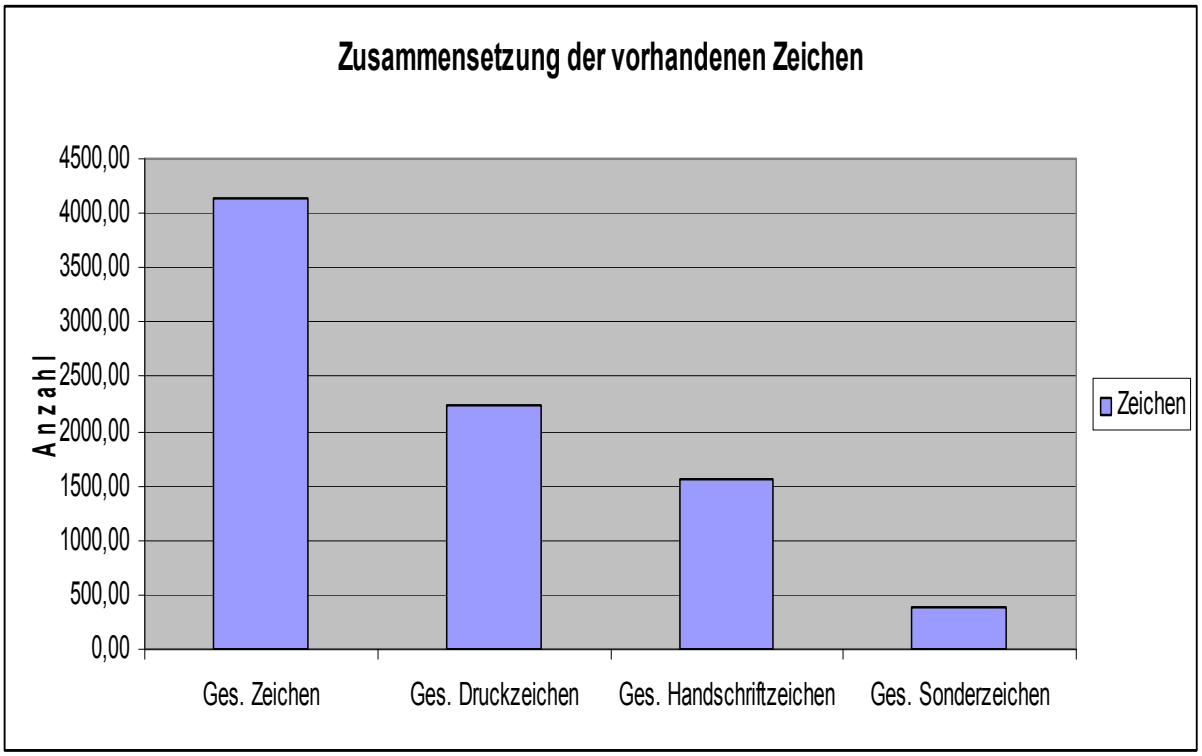

Diagramm 2: Die gesamten Zeichen im absoluten Vergleich

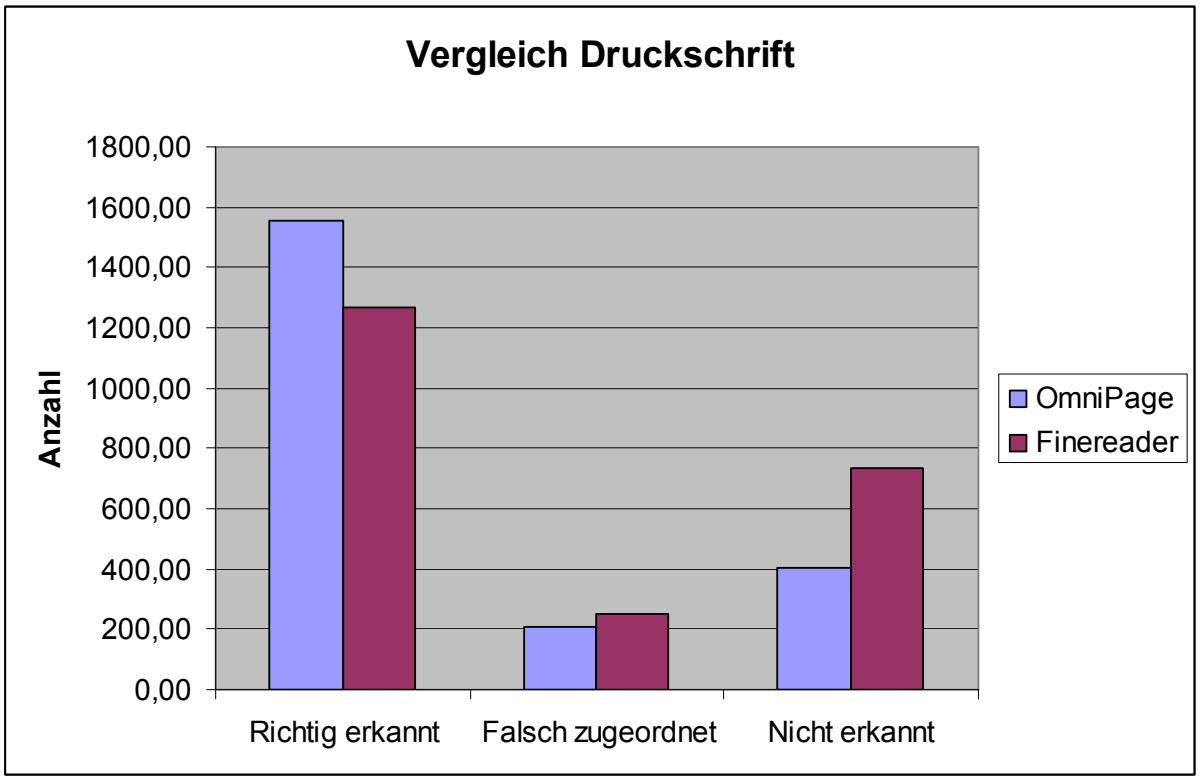

Diagramm 3: Druckschrift-Erkennung Vergleich OmniPage FineReader

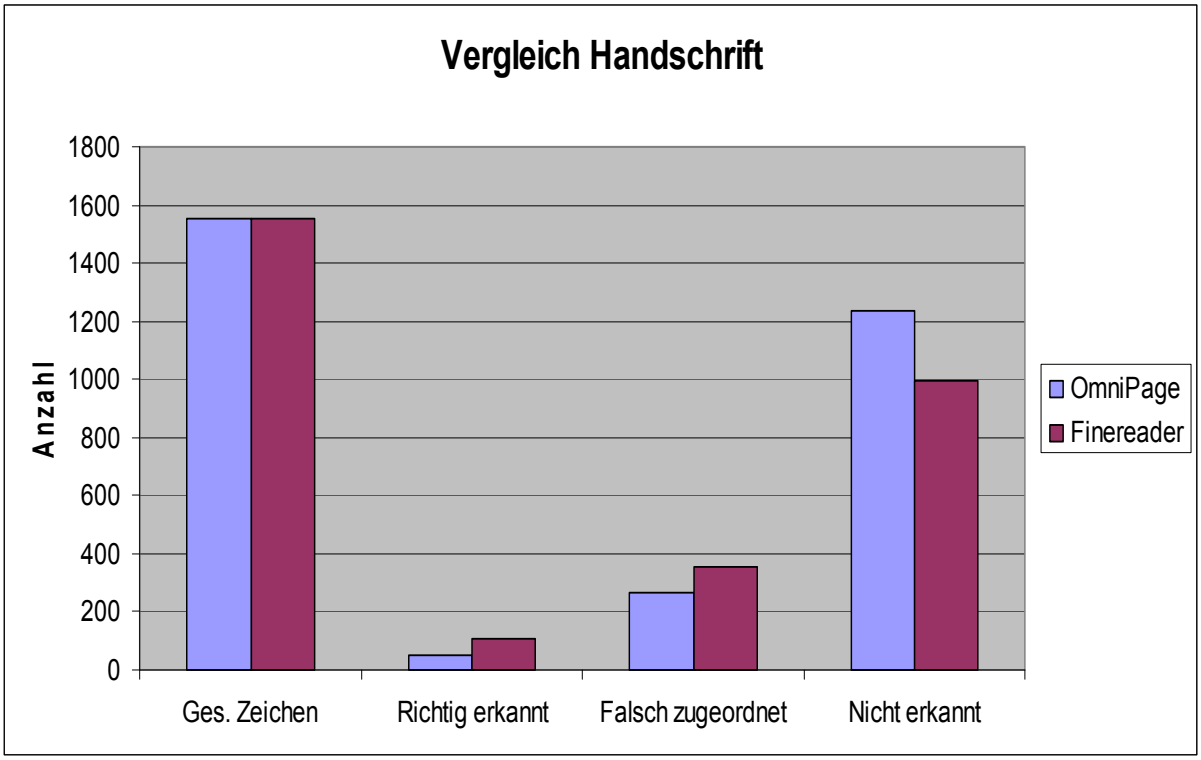

Diagramm 4: Handschrift-Erkennung Vergleich OmniPage FineReader

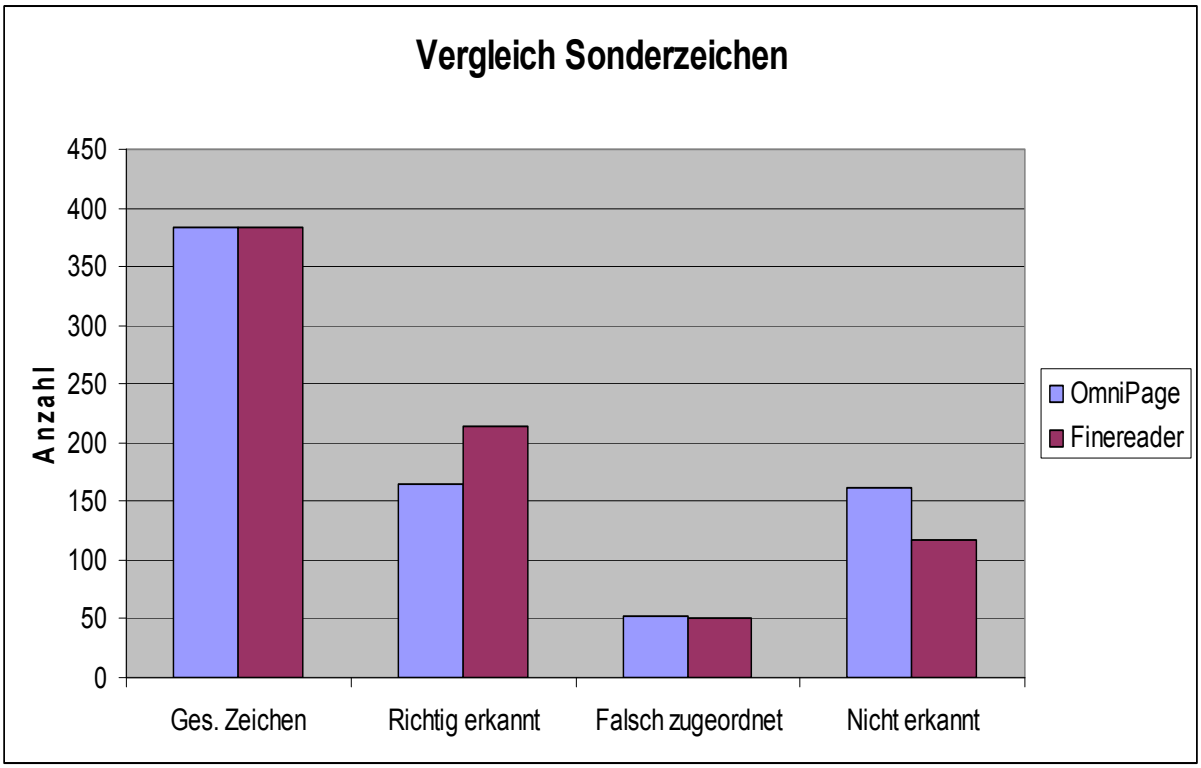

Diagramm 5: Sonderzeichen-Erkennung Vergleich OmniPage FineReader

1.1.7 Bewertung und Ausgabeformat

Die beiden oben genannten Programme sind nicht fähig, handschriftliche Dokumente zu erkennen. Bei Druckschrift liegen sie nah beieinander, ohne dass ein Produkt wirklich entscheidende Vorteile bieten könnte. Im Jahr 2007 haben die beiden Konkurrenten neue Versionen ihrer gut eingeführten OCR-Software herausgebracht. Abbyy liegt beim FineReader jetzt bei der Version 9, während Nuances OmniPage inzwischen schon die Versionsnummer 16 erreicht hat. Wenn man schlechte Vorlagen einscannt, wie bei den Herbarbelegen, ist man mit OmniPage 16 besser beraten, das ein wenig bessere Resultat liefert als der Konkurrent. Zudem kann OmniPage auch besser aus komplexen Umgebungen Textstellen erkennen. Ein wichtiger Vorteil von OmniPage ist der Batchmodus, der große Mengen von Bildern, z.B. nachts, automatisch verarbeiten kann. Das moderne Texterkennungstool erfasst verschiedene Schriftgrößen und erhält das Seitenlayout, um die Vorlage möglichst originalgetreu wiederzugeben. Dazu wird das gängige PDF-Format für die Ausgabe verwendet. Für den Einsatz im Projekt Herbar-Digital scheint zurzeit OmniPage am geeignetesten zu sein. Der erkannte Text kann in verschiedenen Formaten ausgegeben werden.

Abb. 6: Originalbild

Abb. 7: Erkannter Text als PDF-Datei

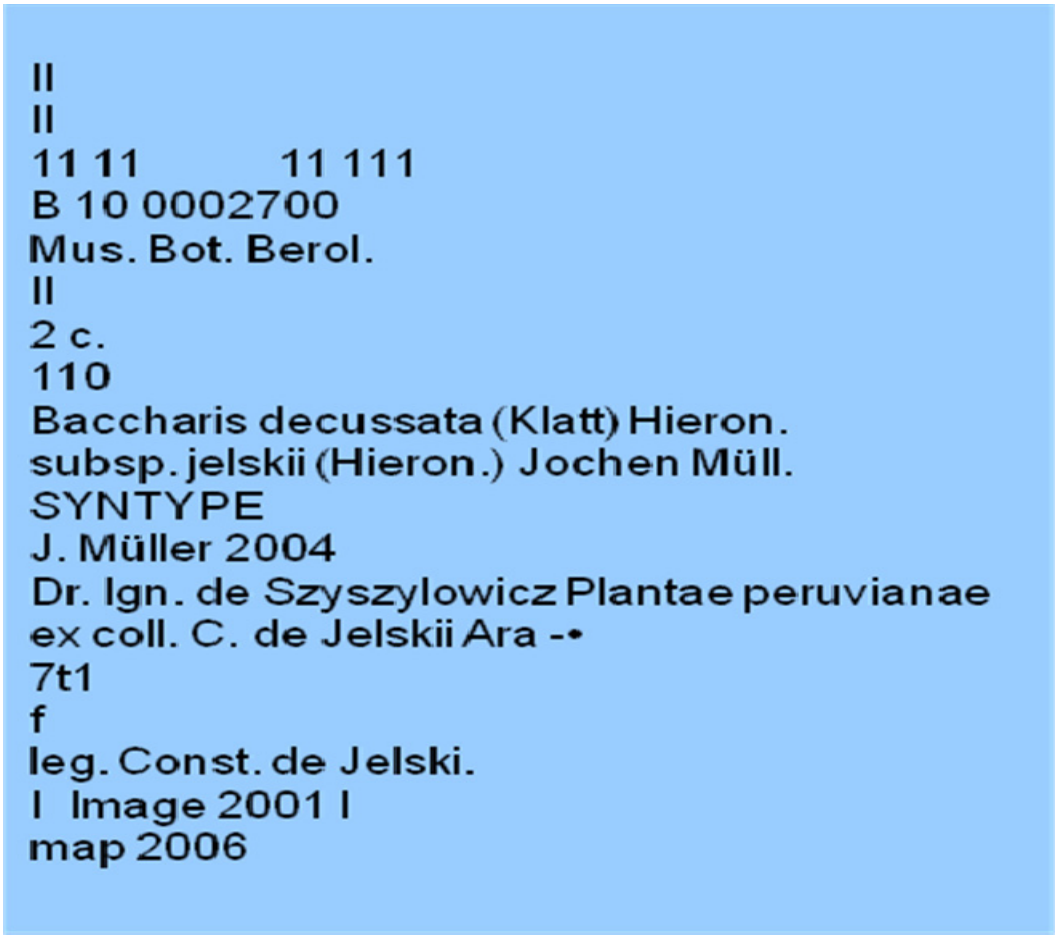

Abb. 8: Erkannter Text als Text-Datei


```
- <run underlined="none" subsuperscript="none"
  fontSize="1100" fontFace="Times New Roman"
  fontFamily="roman" fontPitch="variable"
  spacing="5" foreColor="000000">
    <wd l="5083" t="24019" r="5626"
    b="24182">Const.</wd>
  <space />
</run>
- <run underlined="none" subsuperscript="none"
  fontSize="1100" fontFace="Times New Roman"
  fontFamily="roman" fontPitch="variable"
  spacing="5" foreColor="000000">
    <wd l="5750" t="24029" r="5947"
    b="24182">de</wd>
  <space />
</run>
- <run underlined="none" subsuperscript="none"
  fontSize="1100" fontFace="Times New Roman"
```

```
fontFamily="roman"              fontPitch="variable"
spacing="5" foreColor="000000">
    <wd      l="6067"      t="24029"      r="6662"
    b="24197">Jelski.</wd>
```

Abb. 9: Erkannter Text als XML-Datei (Auszug)

Insbesondere eignet sich die XML-Datei zur Weiterverarbeitung in selbstentwickelter Software, da neben den erkannten Textfragmenten auch Ortskoordinaten angegeben werden. Ein Vergleich mit dem Herbis-System des „Museum of Natural History“ New Haven und New York Botanical Garden zeigt, dass die Erkennungsergebnisse von OmniPage und Herbis vergleichbar sind.

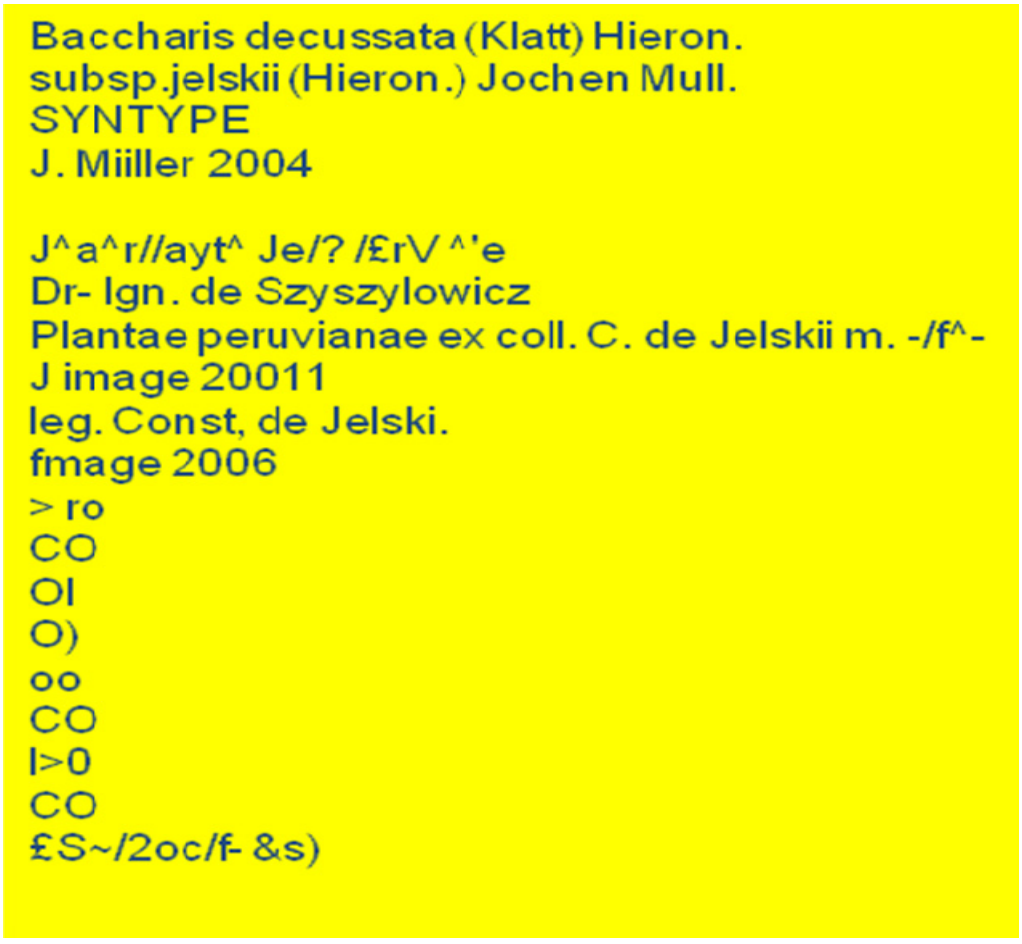

Abb. 10: Erkannter Text des Herbis-Systems

1.2 Untersuchung von kommerzieller Barcodeerkennungs-Software

Da alle Herbarproben mit einem Barcode versehen sind, kann dieser als Primärschlüssel für einen Datenbankeintrag dienen. Der Barcode kann beim Scannen mit einem Barcode-Leser erfasst werden und als Dateiname benutzt werden oder später mit einer Barcodeerkennungs-Software ausgewertet werden und der Dateiname entsprechend geändert werden. Auf diese Weise ist jedes Digitalbild umkehrbar eindeutig einer Pflanzenprobe zugeordnet. Der Barcode ist im Normalfall unten rechts aufgeklebt, kann sich aber auch bei Platzmangel oben rechts oder oben links befinden. Er wird möglichst horizontal aufgeklebt.

1.2.1 Barcodes für Sammlungsobjekte am BGBM

Eine eindeutige Kennzeichnung aller Sammlungsbelege ist aus daten- und verwaltungstechnischer Sicht wünschenswert. Am BGBM hat man sich entschlossen, hierfür einen dem Code-39 Standard entsprechenden Barcode mit Klarschriftzeilen zu benutzen.

Abb. 11: Beispiel für Barcode

Ein festes Format der Kennzeichnung (Länge der verwandten Zahlen oder alphanumerischen Blöcke) ist aus datentechnischen Gründen zu bevorzugen. Die Zeichen (hier: das Zeichen B) vor dem ersten Leerzeichen sollten der Index Herbariorum Abkürzung für die Hauptsammlung entsprechen. Der Index Herbariorum Code wird auch bei anderen Herbarien von einem Leerzeichen gefolgt (es gibt allerdings Ausnahmen, z.B. BM, bei diesen wird aber ein rein numerischer Code verwandt).

1.2.2 Allgemeines Format des BGBM Barcodes

In der ersten Zeile steht in Klarschrift **Mus. Bot. Berol.**, damit die Bögen nicht mehr damit gestempelt werden müssen. Eine Einbeziehung des Akzessionsjahrs in das Barcodeetikett (erste Zeile) würde auch den Akzessionsstempel vermeiden, ist allerdings nur dann möglich, wenn die Etiketten +/- gleichzeitig mit der Benutzung erstellt werden. Die zweite Zeile enthält den Barcode selbst, der in der 3. Zeile in Klarschrift dargestellt wird (ggf. unter Formatierung, z.B. mit Bindestrichen bei der Akzessionsnummer des Botanischen Gartens).
Für den BGBM werden Codes mit einer Gesamtlänge von 12 Zeichen, für Gartenakzessionen und beim Herbarium Willdenow mit 16 Zeichen festgelegt. Der Code beginnt grundsätzlich mit dem Buchstaben B gefolgt von einem Leerzeichen. Um größtmögliche Konsistenz mit vorhandenen Veröffentlichungen und Datensammlungen zu erreichen, soll die bestehende Nummerierung erhalten bleiben, was das Voranstellen eines die Teilsammlung kennzeichnenden Codes notwendig macht. Bei den 12-Zeichen Codes folgt daher eine zweistellige Zahl, die einer Teil- oder Sondersammlung entspricht, gefolgt von einem Leerzeichen und einer 7-stelligen Zeichenfolge, die normalerweise nur aus Ziffern besteht.
Bei Dateinamen werden Leerzeichen durch Unterstriche ersetzt, außerdem steht, wie gewöhnlich, die Dateinamenserweiterung nach einem Punkt (zum Beispiel ".jpg" für JPEG Dateien).
Auf CDs sollte grundsätzlich die 8+3 Namenskonvention verwandt werden, da sonst die Daten eventuell von einigen Betriebssystemen nicht gelesen werden können.
Hier kann eine laufende Nummer verwandt werden, die dann in einer ebenfalls auf der CD befindlichen Textdatei mit dem Verweis auf die Akzessionsnummer versehen wird.

Teilsammlung	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**
Phanerogamenherbar	B		1	#		#	#	#	#	#	#	#				
Holzsammlung	B		1	7		#	#	#	#	#	#	#				
Frucht- und Samen	B		1	8		#	#	#	#	#	#	#				
Samenbank	B		1	9		#	#	#	#	#	#	#				
Farnherbar	B		2	#		#	#	#	#	#	#	#				

Moosherbar	B	3	#		#	#	#	#	#/8	#	#					
Algenherbar	B	4	#		#	#	#	#	#	#	#					
Wassermischproben	B	5	#		#	#	#	#	#	#	#					
Flechtenherbar	B	6	#		#	#	#	#	#	#	#					
Pilzherbar	B	7	#		#	#	#	#	#	#	#					
Nasspräparate	B	8	1		#	#	#	#	#	#	#					
Schaumuseum	B	8	9		#	#	#	#	#	#	#					
Botanischer Garten	B	B	G	#	#	#	#	#	#	#	#	#	#	#	#	
Willdenow-Herbar	B	-	W		9	9	9	9	9	x	-	8	8		7	
Gewebeproben (trocken)	B	G	T		#	#	#	#	#	#	#					

Tabelle 11: Codierungsformate in Teilsammlungen

1.2.3 Barcodeerkennung mit OCR-Programmen

Die Erkennung von Barcodes bei OCR-Programmen beschränkt sich i.Allg. auf die Erkennung der ersten Zeile (s. Abb. 11), in der in Klarschrift **Mus. Bot. Berol.** steht, und dem Barcode der in der 3. Zeile in Klarschrift dargestellt wird.

Anhand von 15 Beispielbildern wird die Erkennungsfähigkeit von OmniPage und FineReader getestet. Es zeigt sich, dass nur waagerechte Barcodes gelesen werden können, aber auch das gelingt nicht immer. Somit sind OCR-Programme für das Lesen von Barcodes ungeeignet.

Bildernummer	Barcode			
	Position	Richtig	Falsch	Keine Erkennung
2b_2011	w	1	0	0
2b_2012	w	1	0	0
2b_2106_Quer	w	1	0	0
2b_2698	w	1	0	0
2b_2699_Quer	w	1	0	0
10_2700_1	w	1	0	0
10_2710_1	s	0	1	0

10_2799_1	s	0	1	0
10_2800_1	w	1	0	0
10_89164_1	s	0	0	1
10_93680_1	w	1	0	0
10_93684_1	w	1	0	0
10_93692_1	w	1	0	0
10_93701_1	w	1	0	0
10_93703_1	w	1	0	0
Gesamt	15	12	2	1
Prozent	100%	80%	13%	7%

Tabelle 12: Barcode-Erkennung OmniPage

Bildernummer	Barcode			
	Position	Richtig	Falsch	Keine Erkennung
2b_2011	w	1	0	0
2b_2012	w	1	0	0
2b_2106_Quer	w	0	0	1
2b_2698	w	1	0	0
2b_2699_Quer	w	0	1	0
10_2700_1	w	1	0	0
10_2710_1	s	0	0	1
10_2799_1	s	0	1	0
10_2800_1	w	1	0	0
10_89164_1	s	0	0	1
10_93680_1	w	1	0	0
10_93684_1	w	1	0	0
10_93692_1	w	1	0	0
10_93701_1	w	0	1	0
10_93703_1	w	1	0	0
Gesamt	15	9	3	3
Prozent	100%	60%	20%	20%

Tabelle 13: Barcode-Erkennung FineReader

1.2.4 Barcodeerkennung mit Barcode-Programmen

Die zweite Zeile (s. Abb. 11) enthält den Barcode selbst, der nur durch spezielle Barcode-Programme zu lesen ist.
Die QS-Barcode Erkennung Version: 4.0 (Barcode Erkennung aus Bild-Dateien) ist eine leistungsfähige Software zur schnellen, automatischen Erkennung von ein- und zweidimensionalen (2D-) Barcodes aus digitalisierten Bildern, die mit Dokumentenscannern, durch Fax und mit Kamerasystemen erzeugt werden. Barcodes sind sehr viel schneller und fehlerfreier zu orten und zu erkennen als Schrift. Für die Barcode-Erkennung werden keine speziellen Barcode-Scanner benötigt. Die QS-Barcode SDK Erkennungssoftware interpretiert das Bild und sucht Barcodes und gibt die Barcode-Inhalte zurück. Es werden zahlreiche Bilddatei-Formate unterstützt. Mit der Software kann die Erkennung der Barcodes in eigene Programme integriert werden. Es werden die üblichen linearen Barcodetypen (Strichcodes) erkannt:

- ☐ Code 39 / erweitert
- ☐ Codabar
- ☐ Code 93 / Code 32
- ☐ Code 2/5 (interleaved, Industrie, etc)
- ☐ EAN 8, EAN 13, UPC A / UPC E

Über Parameter wie Größe, Drehung, Anzahl, Länge des Inhalts, Prüfsummen, Größe, Ruhezone, etc. wird die Erkennung gesteuert. Bei sehr schlecht gedruckten Barcodes kann "Verdacht" gemeldet werden. Es können beliebig viele Barcodes pro Bild erkannt werden.

Der QS-DocumentAssembler ist eine Windows-Anwendung, mit der im Stapelbetrieb massenweise aus Dateien oder in Adobe PDF-Dokumenten Barcodes gelesen werden. Die Bilddateien werden mit Scannern, Kameras oder per Fax erzeugt.
QS-DocumentAssembler verarbeitet alle Dateien aus dem Quellverzeichnis und erkennt die Barcodes darauf. Je nach Programm-Einstellung werden die Dateien umbenannt, die Seiten neu strukturiert oder nur alle Barcodes auf den Seiten gelesen und protokolliert.

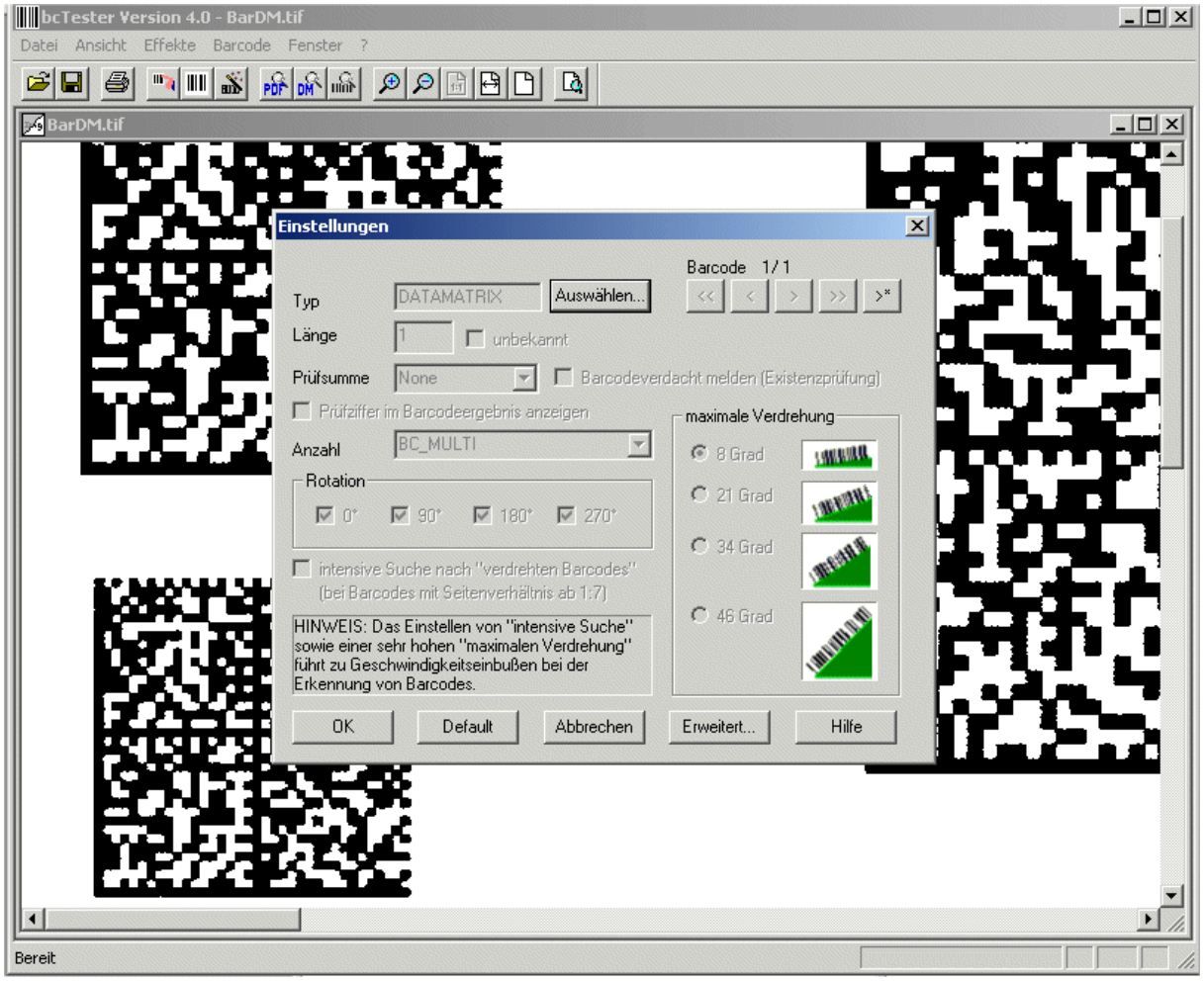

Abb. 12: Die QS-Barcode Erkennung Version: 4.0

Die Software QS-DocumentAssembler wurde im Stapelbetrieb über Nacht auf 465 Originalherbarproben angewendet. Von den 465 Barcodes wurden 415 richtig erkannt. 35 Barcodes wurden nicht gefunden und 15 Barcodes wurden falsch gelesen. Die Erkennungsrate von weniger als 90 % ist natürlich nicht akzeptabel. In Gesprächen mit der Herstellerfirma wurde klar, dass vor der eigentlichen Barcodeerkennung eine Binarisierung durchgeführt wird. Deshalb wurde beschlossen, eigene Verfahren zur Bildvorverarbeitung zu programmieren.

1.2.5 Eigene Verfahren zur Bildvorverarbeitung

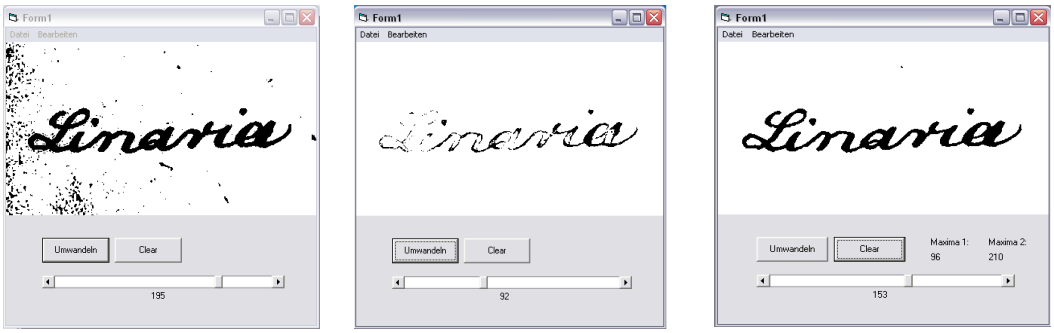

Abb. 13:Schwellenwert zu hoch, zu niedrig, genau richtig

Die oben gezeigten Abbildungen verdeutlichen, welche Probleme bei falsch eingestellten Schwellwerten auftreten können. Im ersten Bild von Abb. 13 ist klar das Hintergrundrauschen zu erkennen, das durch eine nicht gleichmäßig ausgeleuchtete Scanvorlage bzw. ein nicht richtig belichtetes Objekt verursacht wird. Im dritten Bild ist der Schwellwert optimal eingestellt. Schriftprobe und Hintergrund sind gut voneinander getrennt.

Bei ungleichmäßiger Schriftführung bzw. nicht immer ein und derselben Schriftstärke und auch bei unregelmäßiger Beleuchtung kann es passieren, dass man nur Teile von Objekten oder der Schrift sieht und der Rest klar zu erkennen ist, wie im zweiten Bild zu sehen ist. Daher ist es meistens wünschenswert nicht über die gesamte Vorlage eine passende Schwelle zu setzen, sondern verschiedene Bereiche gesondert voneinander zu behandeln. Dabei kann man sich ein adaptives Schwellwertverfahren zunutze machen. Hierbei schaut man sich die Umgebung eines jeden Pixels im Bezug auf seine Nachbarn an, und entscheidet dann ob dieser den Wert 0 oder 1 erhält.

Abb. 14: Menüpunkt Wahl des Schwellwertverfahrens.

Abb. 15: Lokale Umgebung mit bimodalem Histogramm und Schwellwert

In Abb. 15 ist zu erkennen, wie der rot markierte Bereich mit dem gewünschten Schwellwert bearbeitet wurde. Separaten Bereichen können auf diese Art und Weise verschiedene Werte zugeordnet werden. Darüber hinaus besteht die Möglichkeit mittels Brute-Force-Methoden einen Schwellwert zu setzen. Hierbei wird die Schwelle bei einem festgelegten Bruchteil z.B. einem Viertel aller im Histogramm auftretenden Pixel, von links beginnend, gesetzt. Diese Methode ist allerdings nur bei speziellen Gegebenheiten einsetzbar. Mit ihr können recht schnell brauchbare Ergebnisse erzielt werden.

Um ein für die Barcodeerkennung geeigneteres Binärbild herzustellen, wurde nach verschiedenen Tests ein adaptives Schwellwertverfahren ausgewählt. Das Verfahren funktioniert so, dass für jeden Bildpunkt mit Hilfe seiner Umgebung ein Schwellwert berechnet wird, indem vom Mittelwert oder gewichteten Mittelwert eine Konstante abgezogen wird. Das Ergebnis der Binarisierung ist weiß, wenn der aktuelle Grauwert unterhalb dieses Schwellwerts liegt, ansonsten schwarz. Es wurden mit den möglichen Parametern verschiedene Kombinationen ausprobiert und nach einigem Experimentieren gelang es, die Binärbilder so herzustellen, dass mit dem QS-DocumentAssembler eine Erkennungsrate von 100% gelang. Wegen des guten Ergebnisses wurden die Recherchen nach weiterer kommerzieller Barcodeerkennungs-Software zunächst eingestellt.

Abb. 16: Ergebnis der Barcode-Erkennung

Literaturverzeichnis

Dengel, A., Hoch, R., Malburg, M., Weigel, A.: Techniques for Improving OCR Results. Handbook of Character Recognition and Document Image Analysis, S. 227-258. In: Hrg. Bunke, H., Wang, P. S. P., World Scientific Publishing Company,1997.

Dijkstra, E. W. A.: Discipline of Programming, Prentice-Hall Inc., N.J. 1976.

Esakov, J., Lopresti, P., Sandberg, J., Zhou, J.: Issues in Automatic OCR Error Classification. Proceedings of Third Annual Symposium on Document Analysis and Information Retrieval, S. 401-412, Las Vegas Nevada, 1994.

Gusfield, D.: Algorithms on Strings, Trees and Sequences. Cambridge, University Press, 1997.

Ford, G., Hauser, S. E., Le Daniel, X., Thoma, George R.: Pattern matching techniques for correcting low confidence OCR words in a known context. Proceedings of SPIE01, Vol. 4307, Document Recognition and Retrieval VIII, S. 241-249, 2001.

Jones, M. A., Story, G. A., Ballard, B. W.: Integrating multiple knowledge sources in a Bayesian OCR postprocessor. Proceedings of IDCAR-91, St. Malo, France, 1991.

Levenshtein, V. I.: Binary Codes Capable of Correcting Deletions, Insertions and Reversals. Doklady Akademii. Nauk SSSR 163(4): p. 845-848, 1965.

Mori Shunji, Nishida Hirobumi und Yamada Hiromitsu: Optical Character Recognition. John Wiley & Sons, New York 1999.

Schulz, K. U., Mihov S.: Fast String Correction with Levenshtein-Automata. International Journal of Document Analysis and Recognition, 5(1):67{85, 2002.

Wagner, R. A., Fischer, M.: The String to String Correction Problem. Journal ACM 21(1), S. 168-173, 1974.

Weicker, K.: Evolutionäre Algorithmen. Teubner Verlag, Stuttgart, 2002.

Zimmermann, M., Chapplier, J.-C., Bunke, H.: Parsing N-best Lists of Handwritten Sentences Proceedings of ICDAR-03, S. 572-582, Edinburgh, 2003.